03

Clase Funcionario

Transcripción

[00:00] Bueno, vamos a ver que aquí tenemos la estructura básica de cualquier aplicativo recién creado, entonces vamos a darle new class y en esta oportunidad, así como yo ya definí mi clase cuenta, mi clase cliente, voy a definir una nueva clase y esta va a ser funcionario, por ejemplo.

[00:25] ¿Qué es un funcionario? Un funcionario es cualquier trabajador del banco. Entonces le damos finish. Y para mi funcionario yo quiero que él tenga tres atributos, quiero que él tenga un nombre, quiero que él tenga su número de documento y finalmente quiero que bueno, como es un funcionario, tiene que tener un salario, lo más importante.

[01:01] Entonces, como ya hemos aprendido en el curso anterior, vamos a declarar los atributos private y el tipo de dato va a ser string. Al final, punto y coma para que compile y listo. De igual forma con documento. Private string documento, punto y coma, y de igual forma para el salario. Es private, salario va a ser un double salario.

[01:36] Listo. Ya tengo mis tres atributos creados. Ahora vemos que está sombreado en amarillo. ¿Por qué? ¿Qué me dice Eclypse sobre este sombreado en amarillo. Me dice que el valor del campo funcionario no es usado. Este valor no está haciendo uso de ninguna parte del proyecto por lo tanto es código inútil, código inservible y Eclypse me sugiere, lo primero que me sugiere es removerlo.

[02:03] Me dice: "Es código que no estás usando, mejor lo removemos". Podría ser. Y la segunda opción me sale: "Crea un getter y setter por nombre". Entonces, yo voy a tomar esta segunda opción pero no solamente para nombre sino de una vez ya voy a generar getter y setter para todos mis campos.

[02:24] Entonces es solamente clic derecho, source y generate getters and setters. Vamos aquí, seleccionamos qué atributos deseo crear el getter y setter y le doy generate. Y automáticamente Eclypse ya me generó el código que yo necesito para mis getters y setters.

[02:48] Hay una cosa más que yo no estoy aquí declarando y es el constructor. Vamos a recordar un poco. ¿Qué era el constructor? El constructor es aquel método público que el tipo de dato o el tipo de retorno de ese método es la clase en sí, entonces funcionario, no le voy a pasar parámetros y no le asigno ningún valor.

[03:16] Este es mi constructor. Nuevamente a modo de repaso, constructor es un método del mismo tipo de retorno de la clase y que puede aceptar parámetros o no para asignarlos a las variables aquí. Si se dan cuenta, ya ninguna variable está resaltada en amarillo porque ya están siendo usadas aquí abajo en los métodos getter y setter.

[03:42] Entonces, como ya es de costumbre, vamos a crear nuestra clase de pruebas. Voy a subir aquí un poco, new class y le vamos a poner TestFuncionario. Nada nuevo hasta aquí. Entonces ahora vamos a crear nuestro método main. Escribimos main, "Ctrl + espacio", seleccionamos y como ya hemos aprendido bien en el curso anterior, vamos a crear nuestro nuevo funcionario. Le vamos a llamar Diego.

[04:15] New "Ctrl + espacio" Funcionario paréntesis, punto y coma para que compile, y a Diego le vamos a decir lo primero, su nombre va a ser Diego, entonces ponemos Diego, le vamos a asignar un número de documento,

setDocumento. Ese número de documento va a ser este de aquí. Y por último el salario.

[04:52] Listo. Si se dieron cuenta aquí yo no escribí setSalario sino escribí solamente sala, como escribiendo directamente salario. Por ejemplo hago "Ctrl + espacio" y automáticamente Eclypse me dice: "¿quieres getSalario o setSalario?" Yo quiero setSalario. Automáticamente lo selecciono, pongo mi valor, 2000 en este caso, y ya Eclypse me da esa ayuda también.

[05:19] No necesito escribir la firma exacta del método para poder llamarlo. Bueno, ya como estamos acostumbrados vamos a darle un system.out.printIn para ver que esté guardando las informaciones y le vamos a decir que imprima mi salario. Y en este caso atributo es sala y le doy ahora sí un getSalario. Guardamos, botón verde y listo, 2000.

[05:50] Está funcionando. El código compila, funciona, hace lo que debe hacer, hasta ahora nada nuevo.